BAB II
LANDASAN TEORI

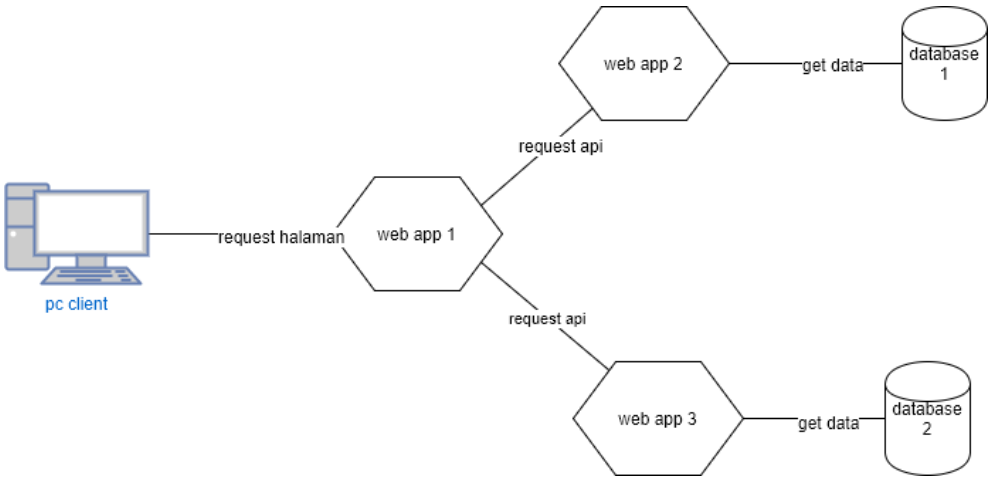
Pada bab ini membahas mengenai teori-teori yang terkait dengan penelitian yang akan dilakukan dan penelitian sebelumnya yang telah dilakukan. Teori-teori tersebut dikumpulkan dari buku, jurnal ataupun *website*.

2.1 Microservice

Microservice adalah sebuah teknologi baru yang sedang ramai diperbincangkan dalam merancang sebuah perangkat lunak sebagai layanan yang dapat digunakan secara independen [8]. Pada dasarnya *Microservice* membuat sebuah aplikasi yang besar dengan cara memecah menjadi service-service yang kecil lalu menggabungkannya agar dapat berjalan secara bersamaan.

Sejarah munculnya *microservice* waktu itu ketika para software arsitek sedang mengadakan pertemuan dan memiliki pandangan yang sama dalam membuat sebuah perangkat lunak. Pada saat tiga hari sebelum berakhirnya pertemenuan tersebut, salah satu dari perserta berteriak dan berkata bagai mana cara membangun sistem yang ada diganti ketika dalam perbaikan? Dan akhirnya muncul istilah micro pada aplikasi tersebut menurut James Lewis[2].

Gambar 2.1 adalah gambaran sederhana *Microservice* dimana setiap aplikasi berjalan sendiri dan berkomunikasi menggunakan REST API. Pada masing-masing aplikasi dapat menggunakan bahasa pemrograman yang berbeda dan menggunakan database yang berbeda sesuai dengan kebutuhan yang akan dibuat oleh developer.



Gambar 2.1 arsitektur *microservice*

2.2 *Website*

Website atau web dapat diartikan sebagai kumpulan halaman-halaman yang digunakan untuk menampilkan informasi teks, gambar diam atau gerak, animasi, suara, dan ataupun gabungan dari semuanya, yang bersifat statis maupun dinamis yang membentuk suatu rangkaian bangunan yang saling terkait, yang masing-masing dihubungkan dengan jaringan-jaringan halaman [10].

2.3 Rest Api

REST (REpresentational State Transfer) merupakan standar arsitektur komunikasi berbasis web yang sering diterapkan dalam pengembangan layanan berbasis web. Umumnya menggunakan HTTP (Hypertext Transfer Protocol) sebagai protocol untuk komunikasi data. REST pertama kali diperkenalkan oleh Roy Fielding pada tahun 2000.

Pada arsitektur REST, REST *server* menyediakan *resource*s (sumber daya/data) dan REST *client* mengakses dan menampilkan *resource* tersebut untuk penggunaan selanjutnya. Setiap *resource* diidentifikasi oleh URIs (Universal *Resource* Identifiers) atau global ID. *Resource* tersebut direpresentasikan dalam bentuk format teks, JSON atau XML. Pada umumnya formatnya menggunakan JSON dan XML [11].

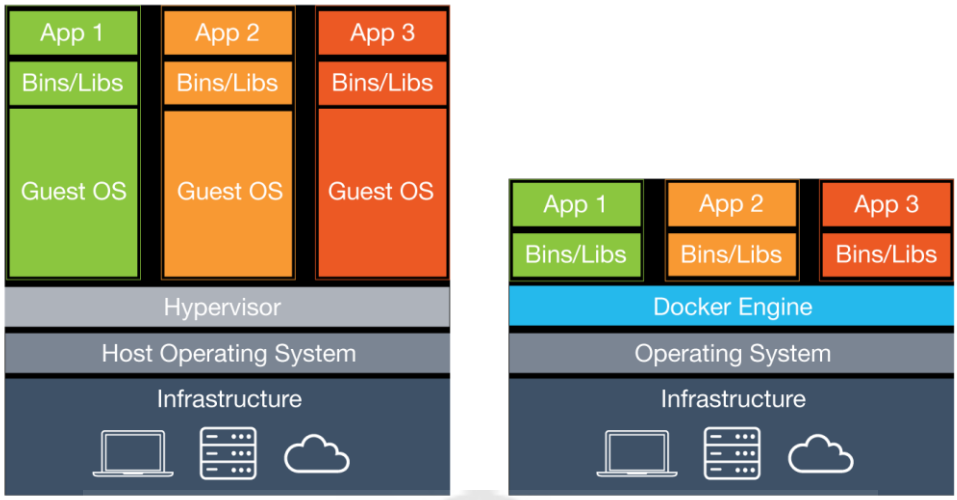
2.4 *Docker*

Docker adalah sebuah teknlogi yang sedang ramai dibahas dalam beberapa penelitian yang berkaitan dengan virtualisasi dan pembuatan web

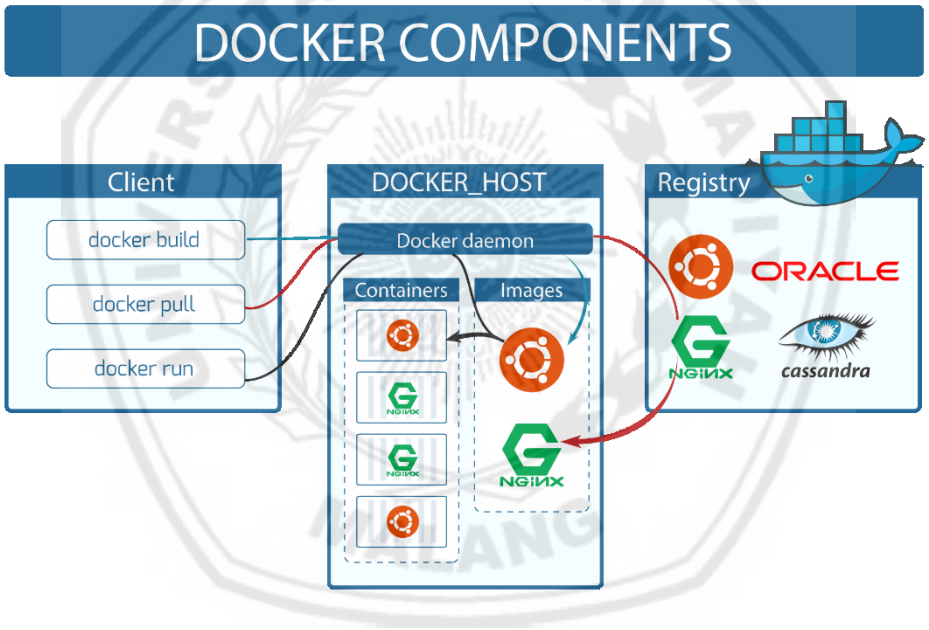
[7]. *Docker* sering digunakan dalam pembuatan aplikasi karena mudah dalam proses pendistribusiannya[5][7]. *Docker* menggunakan teknologi virtualisasi yang dinamakan *container*.

Docker dikembangkan pertama kali oleh Solomon Hykes pada project intenal di DotCloud bersama dengan beberapa koleganya seperti Andrea Luzzardi dan Francois-Xavier Bourlet. *Docker* dirilis pada tahun 2013 silam hingga saat ini masih terus dikembangkan sampai dengan versi 17.05. *Docker* memiliki perbedaan dengan virtualisasi pada umumnya di mana aplikasi berjalan di atas hypervisor dan guest os, sedangkan *docker* dapat menjalankan aplikasi tanpa membutuhkan kedua hal tersebut [12] seperti pada **Gambar 2.2.**

Gambar 2.2 docker memiliki perbedaan dibanding virtualiasasi pada umumnya, dimana docker hanya menggunakan docker engine untuk menjalan aplikasi atau virtualiasasi aplikasi atau lainnya yang menggunakan docker.



Gambar 2.2 perbedaan docker dengan virtualisasi pada umumnya



Gambar 2.3 perbedaan antara docker dan virutalisasi pada umumnya

Gambar 2.3 merupakan gambaran dari arsitektur yang digunakan *docker* dan terdapat beberapa kompen yaitu *docker client*, *docker daemon* dan *docker registry*.

Cara kerja dari *docker* ini adalah membuat image dan menjalankannya menjadi sebuah *container*. *Image* ini dapat diperoleh dari membuat sendiri atau mengambil dari *registry docker*. *Docker* digunakan karena mudah dalam membuat dan mendistribusikan aplikasi karenanya *microservice* sangat cocok

didukung dengan *docker* dalam pembuatannya[5]. Karena *docker* mampu menjalankan aplikasi yang berbeda secara bersamaan serta mudah dalam memonitoring aplikasi yang nantinya menggunakan *microservice*.

2.5 Node.js

Node.js adalah adalah runtime JavaScript yang dibangun di mesin JavaScript V8 Chrome. Node.js menggunakan model I / O event-driven, yang membuatnya ringan dan efisien. Ekosistem paket Node.js, npm merupakan ekosistem perpustakaan open source terbesar di dunia [13]. Node.js adalah termasuk bahasa javascript dimana bahasa pemrograman ini sering digunakan dalam *development* [14].

2.6 PHP

PHP adalah bahasa scripting open source yang umum digunakan yang sangan sesuai untuk pengembangan web dan dapat disematkan ke dalam HTML. Yang membedakan PHP dari javascript *client* adalah kode dieksekusi di *server*, yang menghasilkan HTML yang kemudian dikirim ke *client*. *Client* akan menerima hasil dari menjalankan skrip tersebut, namun tidak akan tahu kode dasarnya [15].

2.7 Python

Python adalah bahasa pemrograman interpretered, berorintasi objek, tingkat tinggi dengan semantik dinamis. Bahasa tingkat tinggi untuk membangun struktur data, dikombinasikan dengan *dynamic typing* dam *dynamic binding,* membuatnya sangat menarik untuk digunakan pada *rapid application development,* dan juga digunakan sebagai bahasa *script* untuk menghubungkan komponen yang ada bersama-sama. Sintaks sederhana python, mudah dipelajari dan memudahkan pembaca, karena hal itu mengurangi biaya pemeliharaan program. Python mendukung modul dan paket yang mendorong modularitas program dan penggunaan kode [16].